DETAILED ACTION

This action is in response to Applicants' amendment filed on 06/28/2011. **Independent Claims 1, 7, 17 and 21** have been amended. Claims 1, 4-7, 10-12, 17, 20-21, 24-25, 27 and 31-40 are currently pending. The applicants' amendments to claims are shown in ***bold and italics,*** and the examiner's response to claim amendments is shown in **bold** in this office action. **This Action is made FINAL.**

Claim Rejections - 35 USC § 112

The following is a quotation of the second paragraph of 35 U.S.C. 112:

> The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

Claims 21, 24, 36 and 40 are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

Claims 21, 24 and 36 include means plus function limitations that invoke 35 U.S.C. 112, sixth paragraph. However, the written description fails to disclose the corresponding structure, material, or acts for the claimed function. The corresponding structure must include the algorithm used to realize the claimed limitations. Applicants' specification does not include the algorithms used.

Claim 40 being dependent on the rejected independent claim 21, inherits all the limitations of the base claim 21, and is therefore rejected for the same reason.

Applicant is required to:

(a) Amend the claim so that the claim limitation will no longer be a means (or step) plus function limitation under 35 U.S.C. 112, sixth paragraph (by deleting all occurrences of the words “means for” from the claim text); or

(b) Amend the written description of the specification such that it expressly recites what structure, material, or acts perform the claimed function without introducing any new matter (35 U.S.C. 132(a)).

If applicant is of the opinion that the written description of the specification already implicitly or inherently discloses the corresponding structure, material, or acts so that one of ordinary skill in the art would recognize what structure, material, or acts perform the claimed function, applicant is required to clarify the record by either:

(a) Amending the written description of the specification such that it expressly recites the corresponding structure, material, or acts for performing the claimed function and clearly links or associates the structure, material, or acts to the claimed function, without introducing any new matter (35 U.S.C. 132(a)); or

(b) Stating on the record what the corresponding structure, material, or acts, which are implicitly or inherently set forth in the written description of the specification, perform the claimed function. For more information, see 37 CFR 1.75(d) and MPEP §§ 608.01(o) and 2181.

Claim Rejections - 35 USC § 103

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

The factual inquiries set forth in *Graham* **v.** *John Deere Co.*, 383 U.S. 1, 148 USPQ 459 (1966), that are applied for establishing a background for determining obviousness under 35 U.S.C. 103(a) are summarized as follows:

1. Determining the scope and contents of the prior art.
2. Ascertaining the differences between the prior art and the claims at issue.
3. Resolving the level of ordinary skill in the pertinent art.
4. Considering objective evidence present in the application indicating obviousness or nonobviousness.

This application currently names joint inventors. In considering patentability of the claims under 35 U.S.C. 103(a), the examiner presumes that the subject matter of the various claims was commonly owned at the time any inventions covered therein were made absent any evidence to the contrary. Applicant is advised of the obligation under 37 CFR 1.56 to point out the inventor and invention dates of each claim that was not commonly owned at the time a later invention was made in order for the examiner to consider the applicability of 35 U.S.C. 103(c) and potential 35 U.S.C. 102(e), (f) or (g) prior art under 35 U.S.C. 103(a).

Claims 1, 7, 17, 21, and 33-40 are rejected under 35 U.S.C. 103(a) as being unpatentable over **Falcon et al.** (**U.S. Patent Publication # 6,295,556 B1**) in view of **Coulthard et al. (U.S. Patent Application Publication # 2004/0003371 A1)** and further in view of **Henriquez (U.S. Patent Application Publication # 2004/0088377**

A1) and further in view of **Perholtz et al. (U.S. Patent Application Publication # 2002/0091850 A1).**

Consider **claim 1**, Falcon et al. show and disclose a user interface for managing a connection between a remote computing device and a local computing device (Fig. 6 that shows a user interface for setting up network and Internet connections; Fig. 7 that shows a second interface to manage connections by setting different configuration options for a connection; column 2, lines 7-9 disclose the same details), comprising:
a desktop at the remote computing device, wherein the desktop is operative to display, using a processor, at least a first connection icon directly on the desktop, the first connection icon representing a first connection between the remote computing device and the first local computing device (Fig. 6 that shows a window over a desktop with a plurality of connection icons named “Office”, “Work From Home” and “MSN” to select from, wherein each connection icon (under the “Connector Name” heading) represents a connection between a server (a first local computing device) and a client/user computer (the remote computing device); column 6, lines 29-43 describe the connection interface in more details),
wherein a user can either select the first connection icon or an active area on the desktop (Fig. 6 that further shows a “New Connector” active area on the desktop window to define a new connection or to select one of the previously defined connection (three of which are shown); column 6, lines 29-38 describe the same details),

wherein selecting the connection icon by the user from the desktop allows a connection represented by the connection icon to become modifiable to alter the connection of the connection icon by the user (Fig. 7 that shows a user interface for receiving configuration information from a user for a selected connection (e.g. configuration information for the “Office” connection shown in Fig. 6); further showing different tabs that allow a user to modify the configuration settings of the selected connection; column 7, lines 1-27 disclose the details of the connection properties that may be modified for each one of the tabs shown in Fig. 7; [Note: This feature is also disclosed by the Henriquez reference shown below),

wherein selecting the active area allows a new connection window to appear (column 6, lines 34-38 which disclose using the “New Connection Wizard” by clicking on the active area labeled “New Connector” to initiate a new connection, then configuring it by supplying property values for the new connection object in Fig. 7), and

upon designating a new connection, allows a second connection icon to be displayed directly on the desktop (Fig. 6 that shows three different connections on the desktop window that were created by the new connection wizard, then displayed as icons in the desktop window; column 6, lines 29-34 describe the same details),

wherein the second connection icon represents a second connection different from the first connection represented by the first connection icon, between the remote computing device and a second local computing device (Fig. 6 that shows a first connection “Office” icon, connecting a client’s computer/workstation with the server on an office

LAN, and a second connection "Work From Home" icon, connecting a client's computer/laptop at home with the server on the office LAN network).

However, Falcon et al. do not specifically disclose a keystroke management window, wherein the keystroke management window is user modifiable to accept a local keystroke management setting, wherein if the local keystroke management setting is enabled, all hotkey keystrokes are processable at the remote computing device, wherein if the local keystroke management setting is disabled, all the hotkey keystrokes are processable at a first local computing device, and ***the remote computing device is configured to ignore*** the hotkey keystrokes, and that the first connection icon is for a first application and the second connection icon is for a second application, wherein the first application is different from the second application; and wherein the desktop is operative to display at least a first application icon directly on the desktop at the remote computing device, wherein the first application icon represents an application available for execution on the first local computing device; wherein the user interface comprises a software feature configured to allow the user to select a connection icon from the desktop of the remote computing device; and wherein the remote computing device includes an operating system which does not allow the connection icon to be modified from the desktop by the user in absence of the software feature.

In the same field of endeavor, Coulthard et al. show and disclose the claimed user interface, wherein a first connection is for a first application and the second connection is for a second application, wherein the first application is different from the second application; and wherein the desktop is operative to display at least a first

application icon directly on the desktop at the remote computing device, wherein the first application icon represents an application available for execution on the first local computing device (Fig. 11, that shows three different connections 1111-1113 between a Developer 1 and three Remote Systems 1120, 1122 and 1124, wherein connection 1111 (the first connection) provides access to Tool A (application 1130) to be executed on the “Remote System 1”, i.e. 1120 (corresponding to the first local computing device, a first server) and connection 1112 (the second connection) provides access to Tool C (application 1150) to be executed on the “Remote System 2”, i.e. 1122 (corresponding to the second local computing device, a second server); paragraph 0099 describes the same details).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to provide a user interface on a desktop, wherein a first connection is for a first application and the second connection is for a second application, and wherein the first application is different from the second application, and wherein the desktop is operative to display at least a first application icon directly on the desktop at the remote computing device, wherein the first application icon represents an application available for execution on the first local computing device, as taught by Coulthard et al., in the user interface of Falcon et al., so as to provide a user a graphical interface to set up and manage network connections based on the needed applications.

However, Falcon et al., as modified by Coulthard et al., do not specifically disclose a keystroke management window, wherein the keystroke management window

is user modifiable to accept a local keystroke management setting, wherein if the local keystroke management setting is enabled, all hotkey keystrokes are processable at the remote computing device, wherein if the local keystroke management setting is disabled, all the hotkey keystrokes are processable at a first local computing device, and ***the remote computing device is configured to ignore*** the hotkey keystrokes, and that wherein the user interface comprises a software feature configured to allow the user to select a connection icon from the desktop of the remote computing device; and wherein the remote computing device includes an operating system which does not allow the connection icon to be modified from the desktop by the user in absence of the software feature.

In the same field of endeavor, Henriquez discloses the claimed user interface, comprising a software feature configured to allow the user to select and modify a connection icon from the desktop of the remote computing device, and wherein the remote computing device includes an operating system which does not allow the connection icon to be modified from the desktop by the user in absence of the software feature (Fig. 2 and paragraph 0036 that describes communication between a local system 210 and a remote system 220, wherein the local system can run an application residing on the remote system, and wherein the remote system sends icon information for the application to the local system, so that when a user selects an icon associated with the remote application on the local system, a remote desktop connection is established; further disclosing that at least a subset of the transmitted iconic connection data is persistent in the memory of the local system, and can be updated with later

format versions, etc.; furthermore paragraphs 0006, 0030, and 0031 disclose that the remote computing device includes a Microsoft® Windows® operating system, e.g. Windows NT SERVER® operating system with Remote Desktop Protocol (RDP) feature); and

wherein selecting the connection icon by the user from the desktop allows a connection represented by the connection icon to become modifiable to alter the connection of the connection icon by the user (Figs. 3 and 6, paragraphs 0045 and 0050 which disclose the claimed user interface, comprising icon data transmitted by a remote system being received by a local system, further disclosing that after decoding, the icon data is streamed into a file and the location of the icon file is noted in a regkey (registry key field), then disclosing that a shell (WINDOWS® operating system) is informed that an icon is ready to be displayed, when a specific file extension (e.g. .ico) is placed on the desktop, thereby disclosing placing the connection icon/link for the transmitted icon on the desktop; and also

in Falcon et al. reference, Figs. 6-7 and column 7, lines 1-27 which show and disclose that when a user clicks on to select one of the connection icons (shown in Fig. 6 as “Office”, “Work from Home”, “MSN”, and “New Connector”), a user interface with various tabs to modify the selected existing connection or to enter configuration data for a new connection is displayed, as shown in Fig. 7).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to provide a user interface that comprises a software feature configured to allow the user to select a connection icon from the desktop of the

remote computing device, wherein selecting the connection icon by the user from the desktop allows a connection represented by the connection icon to become modifiable to alter the connection of the connection icon by the user, and wherein the remote computing device includes an operating system which does not allow the connection icon to be modified from the desktop by the user to alter the connection of the connection icon in absence of the software feature, as taught by Henriquez, in the user interface of Falcon et al., as modified by Coulthard et al., because such an operating system will provide a user-friendly interface for remote desktop applications, allowing users to easily modify the connection to applications executable on remote computers.

However, Falcon et al., as modified by Coulthard et al. and Henriquez, do not specifically disclose a keystroke management window, wherein the keystroke management window is user modifiable to accept a local keystroke management setting, wherein if the local keystroke management setting is enabled, all hotkey keystrokes are processable at the remote computing device, wherein if the local keystroke management setting is disabled, all the hotkey keystrokes are processable at a first local computing device, and ***the remote computing device is configured to ignore*** the hotkey keystrokes.

In the same field of endeavor, Perholtz et al. disclose a user interface, further comprising a keystroke management window, wherein the keystroke management window is user modifiable to accept a local keystroke management setting, wherein if the local keystroke management setting is enabled, all hotkey keystrokes are processable at the remote computing device, wherein if the local keystroke

management setting is disabled, all the hotkey keystrokes are processable at a first local computing device, and ***the remote computing device is configured to ignore*** the hotkey keystrokes (Flowchart of Fig. 7G, decision block 759 that checks for use of "Hot Keys"; paragraph 0288, lines 1-16 that disclose the use of "Hot Keys" for redirecting remote client's input keystrokes/mouse data to the local host [i.e. by selecting, using hotkey keystrokes or mouse-clicks, from the Host menu, the option "Exit to Host mode", that disables the local keystroke management setting, such that all the hotkey keystrokes are processed at the local computing device instead of at the remote computing device] and means to return back to the remote client's normal mode of operation by tapping the left shift key three times within 2 seconds [a hotkey sequence that enables the local keystroke management setting, such that all the hotkey keystrokes are now processed back at the remote computing device]).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to provide a user interface, further comprising a keystroke management window, wherein the keystroke management window is user modifiable to accept a local keystroke management setting, wherein if the local keystroke management setting is enabled, all hotkey keystrokes are processable at the remote computing device, wherein if the local keystroke management setting is disabled, all the hotkey keystrokes are processable at a first local computing device, and the remote computing device is configured to ignore the hotkey keystrokes, as taught by Perholtz et al., in the user interface of Falcon et al., as modified by Coulthard et al. and Henriquez, so as to provide a user ability to use hot keys to execute

applications at the local computing device as well as at the remote computing device, and be able to easily switch between them.

Consider **claim 7**, Falcon et al. show and disclose a method for managing a connection between a local computing device and a remote computing device using a user interface (Fig. 6 that shows a user interface for setting up network and Internet connections; Fig. 7 that shows a second interface to manage connections by setting different configuration options for a connection; column 2, lines 7-9 disclose the same details), comprising the steps of:

displaying a desktop at the remote computing device (Fig. 6 that displays a desktop with a user interface for setting up network and Internet connections at the remote computing device);

displaying at least a first connection icon directly on the desktop, the first connection icon representing a first connection between the remote computing device and the first local computing device (Fig. 6 that shows a window over a desktop with a plurality of connection icons named “Office”, “Work From Home” and “MSN” to select from, wherein each connection icon (under the “Connector Name” heading) represents a connection between a server (a first local computing device) and a client/user computer (the remote computing device); column 6, lines 29-43 describe the connection interface in more details);

wherein selecting the connection icon by the user from the desktop allows a connection represented by the connection icon to become modifiable to alter the connection of the

connection icon by the user (Fig. 7 that shows a user interface for receiving configuration information from a user for a selected connection (e.g. configuration information for the "Office" connection shown in Fig. 6); further showing different tabs that allow a user to modify the configuration settings of the selected connection; column 7, lines 1-27 disclose the details of the connection properties that may be modified for each one of the tabs shown in Fig. 7; [Note: This feature is also disclosed by the Henriquez reference shown below]),

receiving a user selection of an active area of the desktop (Fig. 6 that further shows a "New Connector" active area on the desktop window to define a new connection; column 6, lines 29-38 describe the same details);

wherein the user selection of the active area allows a second connection icon to be displayed directly on the desktop, wherein the second connection icon represents a second connection different than the first connection represented by the first connection icon (Fig. 6 that shows a second connection "Work From Home" icon, connecting a client's computer/laptop at home with the server on the Office LAN network, which is different than a first connection (shown as "Office" in Fig. 6)).

However, Falcon et al. do not specifically disclose a keystroke management window, wherein the keystroke management window is user modifiable to accept a local keystroke management setting, wherein if the local keystroke management setting is enabled, all hotkey keystrokes are processed at the remote computing device, wherein if the local keystroke management setting is disabled, all the hotkey keystrokes are processed at a first local computing device, and the hotkey keystrokes ***are ignored*** at

the remote computing device, and that the first connection icon is for a first application and the second connection icon is for a second application; displaying at least a first application icon on the desktop at the remote computing device, wherein the first application icon represents an application available for execution on the first local computing device; wherein the user interface comprises a software feature configured to allow a user to select a connection icon from the desktop of the remote computing device; and wherein the remote computing device includes an operating system which does not allow the connection of the connection icon to be modified from the desktop by the user in absence of the software feature.

In the same field of endeavor, Coulthard et al. show and disclose the claimed method, wherein a first connection is for a first application and the second connection is for a second application, and displaying at least a first application icon on the desktop at the remote computing device, wherein the first application icon represents an application available for execution on the first local computing device (Fig. 11, that shows three different connections 1111-1113 between a Developer 1 and three Remote Systems 1120, 1122 and 1124, wherein connection 1111 (the first connection) provides access to Tool A (application 1130) to be executed on the “Remote System 1”, i.e. 1120 (corresponding to the first local computing device, a first server) and connection 1112 (the second connection) provides access to Tool C (application 1150) to be executed on the “Remote System 2”, i.e. 1122 (corresponding to the second local computing device, a second server); paragraph 0099 describes the same details).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to provide a method, wherein a first connection is for a first application and the second connection is for a second application, and displaying at least a first application icon on the desktop at the remote computing device, wherein the first application icon represents an application available for execution on the first local computing device, as taught by Coulthard et al., in the method of Falcon et al., so as to provide a user a graphical interface to set up and manage network connections based on the needed applications.

However, Falcon et al., as modified by Coulthard et al., do not specifically disclose a keystroke management window, wherein the keystroke management window is user modifiable to accept a local keystroke management setting, wherein if the local keystroke management setting is enabled, all hotkey keystrokes are processed at the remote computing device, wherein if the local keystroke management setting is disabled, all the hotkey keystrokes are processed at a first local computing device, and the hotkey keystrokes ***are ignored*** at the remote computing device, and that the user interface for the method comprises a software feature configured to allow a user to select a connection icon from the desktop of the remote computing device; and wherein the remote computing device includes an operating system which does not allow the connection icon to be modified from the desktop by the user in absence of the software feature.

In the same field of endeavor, Henriquez discloses the claimed method, wherein the user interface comprises a software feature configured to allow the user to select a

connection icon from the desktop of the remote computing device, wherein the remote computing device includes an operating system which does not allow the connection icon to be modified from the desktop by the user in absence of the software feature (Fig. 2 and paragraph 0036 that describes communication between a local system 210 and a remote system 220, wherein the local system can run an application residing on the remote system, and wherein the remote system sends icon information for the application to the local system, so that when a user selects an icon associated with the remote application on the local system, a remote desktop connection is established; further disclosing that at least a subset of the transmitted iconic connection data is persistent in the memory of the local system, and can be updated with later format versions, etc.; furthermore paragraphs 0006, 0030, and 0031 disclose that the remote computing device includes a Microsoft® Windows® operating system, e.g. Windows NT SERVER® operating system with Remote Desktop Protocol (RDP) feature); and wherein selecting the connection icon by the user from the desktop allows a connection represented by the connection icon to become modifiable to alter the connection of the connection icon by the user (Figs. 3 and 6, paragraphs 0045 and 0050 which disclose the claimed method, comprising icon data transmitted by a remote system being received by a local system, further disclosing that after decoding, the icon data is streamed into a file and the location of the icon file is noted in a regkey (registry key field), then disclosing that a shell (WINDOWS® operating system) is informed that an icon is ready to be displayed, when a specific file extension (e.g. .ico) is placed on the

desktop, thereby disclosing placing the connection icon/link for the transmitted icon on the desktop, and also

in Falcon et al. reference, Figs. 6-7 and column 7, lines 1-27 which show and disclose that when a user clicks on to select one of the connection icons (shown in Fig. 6 as "Office", "Work from Home", "MSN", and "New Connector"), a user interface with various tabs to modify the selected existing connection or to enter configuration data for a new connection is displayed, as shown in Fig. 7).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to provide a software feature configured to allow the user to select a connection icon from the desktop of the remote computing device, wherein selecting the connection icon by the user from the desktop allows a connection represented by the connection icon to become modifiable to alter the connection of the connection icon by the user, and wherein the remote computing device includes an operating system which does not allow the connection icon to be modified from the desktop by the user to alter the connection of the connection icon in absence of the software feature, as taught by Henriquez, in the method of Falcon et al., as modified by Coulthard et al., because such a method will provide a user-friendly interface for remote desktop applications, allowing users to easily modify the connection to applications executable on remote computers.

However, Falcon et al., as modified by Coulthard et al. and Henriquez, do not specifically disclose a keystroke management window, wherein the keystroke management window is user modifiable to accept a local keystroke management

setting, wherein if the local keystroke management setting is enabled, all hotkey keystrokes are processed at the remote computing device, wherein if the local keystroke management setting is disabled, all the hotkey keystrokes are processed at a first local computing device, and the hotkey keystrokes ***are ignored*** at the remote computing device.

In the same field of endeavor, Perholtz et al. disclose the claimed method, further comprising displaying a keystroke management window, wherein the keystroke management window is user modifiable to accept a local keystroke management setting, wherein if the local keystroke management setting is enabled, all hotkey keystrokes are processed at the remote computing device, wherein if the local keystroke management setting is disabled, all the hotkey keystrokes are processed at a first local computing device, and the hotkey keystrokes ***are ignored*** at the remote computing device (Flowchart of Fig. 7G, decision block 759 that checks for use of “Hot Keys”; paragraph 0288, lines 1-16 that disclose the use of “Hot Keys” for redirecting remote client’s input keystrokes/mouse data to the local host [i.e. by selecting, using hotkey keystrokes or mouse-clicks, from the Host menu, the option “Exit to Host mode”, that disables the local keystroke management setting, such that all the hotkey keystrokes are processed at the local computing device instead of at the remote computing device] and means to return back to the remote client’s normal mode of operation by tapping the left shift key three times within 2 seconds [a hotkey sequence that enables the local keystroke management setting, such that all the hotkey keystrokes are now processed back at the remote computing device]).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to provide a user interface, further comprising a keystroke management window, wherein the keystroke management window is user modifiable to accept a local keystroke management setting, wherein if the local keystroke management setting is enabled, all hotkey keystrokes are processed at the remote computing device, wherein if the local keystroke management setting is disabled, all the hotkey keystrokes are processed at a first local computing device, and the hotkey keystrokes are ignored at the remote computing device, as taught by Perholtz et al., in the method of Falcon et al., as modified by Coulthard et al. and Henriquez, so as to provide a user ability to use hot keys to execute applications at the local computing device as well as at the remote computing device, and be able to easily switch between them.

Consider **claim 17**, Falcon et al. disclose a non-transitory computer readable medium ***encoded with computer-executable program code, said computer-executable program code*** for managing a connection between a local computing device and a remote computing device using a user interface (claim 11; Fig. 6 that shows a user interface for setting up network and Internet connections; Fig. 7 that shows a second interface to manage connections by setting different configuration options for a connection; column 2, lines 7-9 disclose the same details), the computer-executable program code comprising:

code for displaying a desktop at the remote computing device (Fig. 6 that displays a desktop with a user interface for setting up network and Internet connections at the remote computing device);

code for displaying at least a first connection icon directly on the desktop, the first connection icon representing a first connection between the remote computing device and a first local computing device (Fig. 6 that shows a window over a desktop with a plurality of connection icons named “Office”, “Work From Home” and “MSN” to select from, wherein each connection icon (under the “Connector Name” heading) represents a connection between a server (a first local computing device) and a client/user computer (the remote computing device); column 6, lines 29-43 describe the connection interface in more details);

wherein selecting the connection icon by the user from the desktop allows a connection represented by the connection icon to become modifiable to alter the connection of the connection icon by the user (Fig. 7 that shows a user interface for receiving configuration information from a user for a selected connection (e.g. configuration information for the “Office” connection shown in Fig. 6); further showing different tabs that allow a user to modify the configuration settings of the selected connection; column 7, lines 1-27 disclose the details of the connection properties that may be modified for each one of the tabs shown in Fig. 7; [Note: This feature is also disclosed by the Henriquez reference shown below]);

code for receiving a user selection of an active area of the desktop (Fig. 6 that further shows a "New Connector" active area on the desktop to define a new connection; column 6, lines 29-38 describe the same details),

wherein the user selection of the active area allows a second connection icon to be displayed directly on the desktop (Fig. 6 that shows three different connections on the desktop window that were created by the new connection wizard, then displayed as icons on the desktop window; column 6, lines 29-34 describe the same details),

wherein the second connection icon represents a second connection different than the first connection represented by the first connection icon (Fig. 6 that shows a first connection "Office" icon, connecting a client's computer/workstation with the server on an office LAN, and a second connection "Work From Home" icon, connecting a client's computer/laptop at home with the server on the Office LAN network).

However, Falcon et al. do not specifically disclose code for displaying a keystroke management window, wherein the keystroke management window is user modifiable to accept a local keystroke management setting, wherein if the local keystroke management setting is enabled, all hotkey keystrokes are processable at the remote computing device, wherein if the local keystroke management setting is disabled, all the hotkey keystrokes are processable at a first local computing device, and ***the remote computing device is configured to ignore*** the hotkey keystrokes, and that the first connection icon is for a first application and the second connection icon is for a second application, and code for displaying at least a first application icon on the desktop at the remote computing device, wherein the first application icon represents an

application available for execution on the first local computing device; wherein the user interface comprises a software feature configured to allow the user to select and modify a connection icon from the desktop of the remote computing device; and wherein the remote computing device includes an operating system which does not allow the connection of the connection icon to be modified from the desktop by the user in absence of the software feature.

In the same field of endeavor, Coulthard et al. disclose the claimed non-transitory computer readable medium, wherein a first connection is for a first application and the second connection is for a second application, and code for displaying at least a first application icon on the desktop at the remote computing device, wherein the first application icon represents an application available for execution on the first local computing device (Fig. 11, that shows three different connections 1111-1113 between a Developer 1 and three Remote Systems 1120, 1122 and 1124, wherein connection 1111 (the first connection) provides access to Tool A (application 1130) to be executed on the “Remote System 1”, i.e. 1120 (corresponding to the first local computing device, a first server) and connection 1112 (the second connection) provides access to Tool C (application 1150) to be executed on the “Remote System 2”, i.e. 1122 (corresponding to the second local computing device, a second server); paragraph 0099 describes the same details).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to provide a computer-executable program code stored on a non-transitory computer readable medium, wherein a first connection is for a first

application and the second connection is for a second application, and code for displaying at least a first application icon on the desktop at the remote computing device, wherein the first application icon represents an application available for execution on the first local computing device, as taught by Coulthard et al., in the non-transitory computer readable medium of Falcon et al., so as to provide a user with the executable program code to set up and manage network connections based on the needed applications.

However, Falcon et al., as modified by Coulthard et al., do not specifically disclose code for displaying a keystroke management window, wherein the keystroke management window is user modifiable to accept a local keystroke management setting, wherein if the local keystroke management setting is enabled, all hotkey keystrokes are processable at the remote computing device, wherein if the local keystroke management setting is disabled, all the hotkey keystrokes are processable at a first local computing device, and ***the remote computing device is configured to ignore*** the hotkey keystrokes, and that the code for the user interface comprises a software feature configured to allow the user to select and modify a connection icon from the desktop of the remote computing device, and wherein the remote computing device includes an operating system which does not allow the connection icon to be modified from the desktop by the user in absence of the software feature.

In the same field of endeavor, Henriquez discloses the claimed computer-executable program code stored on a non-transitory computer readable medium, comprising, in the user interface, a software feature configured to allow the user to

select a connection icon from the desktop of the remote computing device, wherein the remote computing device includes an operating system which does not allow the connection icon to be modified from the desktop by the user in absence of the software feature (Fig. 2 and paragraph 0036 that describes communication between a local system 210 and a remote system 220, wherein the local system can run an application residing on the remote system, and wherein the remote system sends icon information for the application to the local system, so that when a user selects an icon associated with the remote application on the local system, a remote desktop connection is established; further disclosing that at least a subset of the transmitted iconic connection data is persistent in the memory of the local system, and can be updated with later format versions, etc.; furthermore paragraphs 0006, 0030, and 0031 disclose that the remote computing device includes a Microsoft® Windows® operating system, e.g. Windows NT SERVER® operating system with Remote Desktop Protocol (RDP) feature)*;* and

wherein selecting the connection icon by the user from the desktop allows a connection represented by the connection icon to become modifiable to alter the connection of the connection icon by the user (Figs. 3 and 6, paragraphs 0045 and 0050 which disclose the claimed method, comprising icon data transmitted by a remote system being received by a local system, further disclosing that after decoding, the icon data is streamed into a file and the location of the icon file is noted in a regkey (registry key field), then disclosing that a shell (WINDOWS® operating system) is informed that an icon is ready to be displayed, when a specific file extension (e.g. .ico) is placed on the

desktop, thereby disclosing placing the connection icon/link for the transmitted icon on the desktop, and

in Falcon et al. reference, Figs. 6-7 and column 7, lines 1-27 which show and disclose that when a user clicks on to select one of the connection icons (shown in Fig. 6 as “Office”, “Work from Home”, “MSN”, and “New Connector”), a user interface with various tabs to modify the selected existing connection or to enter configuration data for a new connection is displayed, as shown in Fig. 7).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to provide, in the computer-executable program code stored on a non-transitory computer readable medium, a software feature configured to allow the user to select a connection icon from the desktop of the remote computing device, wherein selecting the connection icon by the user from the desktop allows a connection represented by the connection icon to become modifiable to alter the connection of the connection icon by the user, and wherein the remote computing device includes an operating system which does not allow the connection icon to be modified from the desktop by the user to alter the connection of the connection icon in absence of the software feature, as taught by Henriquez, in the non-transitory computer readable medium of Falcon et al., as modified by Coulthard et al., because such a method will provide a user-friendly interface for remote desktop applications, allowing users to easily modify the connection to applications executable on remote computers.

However, Falcon et al., as modified by Coulthard et al. and Henriquez, do not specifically disclose code for displaying a keystroke management window, wherein the

keystroke management window is user modifiable to accept a local keystroke management setting, wherein if the local keystroke management setting is enabled, all hotkey keystrokes are processable at the remote computing device, wherein if the local keystroke management setting is disabled, all the hotkey keystrokes are processable at a first local computing device, and ***the remote computing device is configured to ignore*** the hotkey keystrokes.

In the same field of endeavor, Perholtz et al. disclose the computer-executable program code stored on a non-transitory computer readable medium, further comprising code for displaying a keystroke management window, wherein the keystroke management window is user modifiable to accept a local keystroke management setting, wherein if the local keystroke management setting is enabled, all hotkey keystrokes are processable at the remote computing device, wherein if the local keystroke management setting is disabled, all the hotkey keystrokes are processable at a first local computing device, and ***the remote computing device is configured to ignore*** the hotkey keystrokes (Flowchart of Fig. 7G, decision block 759 that checks for use of “Hot Keys”; paragraph 0288, lines 1-16 that disclose the use of “Hot Keys” for redirecting remote client’s input keystrokes/mouse data to the local host [i.e. by selecting, using hotkey keystrokes or mouse-clicks, from the Host menu, the option “Exit to Host mode”, that disables the local keystroke management setting, such that all the hotkey keystrokes are processed at the local computing device instead of at the remote computing device] and means to return back to the remote client’s normal mode of operation by tapping the left shift key three times within 2 seconds [a hotkey sequence

that enables the local keystroke management setting, such that all the hotkey keystrokes are now processed back at the remote computing device]).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to provide code for displaying a keystroke management window, wherein the keystroke management window is user modifiable to accept a local keystroke management setting, wherein if the local keystroke management setting is enabled, all hotkey keystrokes are processable at the remote computing device, wherein if the local keystroke management setting is disabled, all the hotkey keystrokes are processable at a first local computing device, and the remote computing device is configured to ignore the hotkey keystrokes, as taught by Perholtz et al., in the computer-executable program code stored on a non-transitory computer readable medium of Falcon et al., as modified by Coulthard et al. and Henriquez, so as to provide a user ability to use hot keys to execute applications at the local computing device as well as at the remote computing device, and be able to easily switch between them.

Consider **claim 21**, Falcon et al. show and disclose a programmed computer apparatus for managing a connection between a local computing device and a remote computing device using a user interface (Fig. 6 that shows a computer desktop with a user interface for setting up network and Internet connections; Fig. 7 that shows a computer desktop with a user interface to manage connections by setting different

configuration options for a connection; column 2, lines 7-9 disclose the same details), said programmed computer apparatus comprising:

means for displaying a desktop at the remote computing device (Fig. 6 that displays a computer desktop with a user interface for setting up network and Internet connections at the remote computing device),

means for displaying, using a processor, at least a first connection icon directly on the desktop, the first connection icon representing a first connection between the remote computing device and a first local computing device (Fig. 6 that shows a window over a desktop on a computer with a plurality of connection icons named “Office”, “Work From Home” and “MSN” to select from, wherein each connection icon (under the “Connector Name” heading) represents a connection between a server (a first local computing device) and a client/user computer (the remote computing device); column 6, lines 29-43 describe the claimed apparatus in more details);

wherein selecting the connection icon by the user from the desktop allows a connection represented by the connection icon to become modifiable to alter the connection of the connection icon by the user (Fig. 7 that shows a user interface for receiving configuration information from a user for a selected connection (e.g. configuration information for the “Office” connection shown in Fig. 6); further showing different tabs that allow a user to modify the configuration settings of the selected connection; column 7, lines 1-27 disclose the details of the connection properties that may be modified for each one of the tabs shown in Fig. 7; [Note: This feature is also disclosed by the Henriquez reference shown below]);

means for receiving a user selection of an active area of the desktop, wherein the user selection of the active area allows a second connection icon for a second application to be displayed directly on the desktop (Fig. 6 that shows a “New Connector” active area on the desktop window to define a new connection; Fig. 6 further shows three different connections on the desktop window that were created by the new connection wizard, then displayed as icons on the desktop window; column 6, lines 29-38 describe the same details);

wherein the second connection icon represents a second connection different than the first connection represented by the first connection icon (Fig. 6 that shows a first connection “Office” icon, connecting a client’s computer/workstation with the server on an office LAN, and a second connection “Work From Home” icon, connecting a client’s computer/laptop at home with the server on the Office LAN network).

However, Falcon et al. do not specifically disclose means for displaying a keystroke management window, wherein the keystroke management window is user modifiable to accept a local keystroke management setting, wherein if the local keystroke management setting is enabled, all hotkey keystrokes are processable at the remote computing device, wherein if the local keystroke management setting is disabled, all the hotkey keystrokes are processable at a first local computing device, and ***the remote computing device is configured to ignore*** the hotkey keystrokes, and that the first connection icon is for a first application and the second connection icon is for a second application, and means for displaying at least a first application icon on the desktop at the remote computing device, wherein the first application icon

represents an application available for execution on the first local computing device; wherein the user interface comprises a software feature configured to allow the user to select a connection icon from the desktop of the remote computing; and wherein the remote computing device includes an operating system which does not allow the connection of the connection icon to be modified from the desktop by the user in absence of the software feature.

In the same field of endeavor, Coulthard et al. show and disclose the claimed programmed computer apparatus, wherein a first connection is for a first application and the second connection is for a second application, and means for displaying at least a first application icon on the desktop at the remote computing device, wherein the first application icon represents an application available for execution on the first local computing device (Fig. 11, that shows three different connections 1111-1113 between a Developer 1 and three Remote Systems 1120, 1122 and 1124, wherein connection 1111 (the first connection) provides access to Tool A (application 1130) to be executed on the “Remote System 1”, i.e. 1120 (corresponding to the first local computing device, a first server) and connection 1112 (the second connection) provides access to Tool C (application 1150) to be executed on the “Remote System 2”, i.e. 1122 (corresponding to the second local computing device, a second server); paragraph 0099 describes the same details).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to provide a programmed computer apparatus, wherein a first connection is for a first application and the second connection is for a

second application, and means for displaying at least a first application icon on the desktop at the remote computing device, wherein the first application icon represents an application available for execution on the first local computing device, as taught by Coulthard et al., in the programmed computer apparatus of Falcon et al., so as to provide a user an apparatus to set up and manage network connections based on the needed applications.

However, Falcon et al., as modified by Coulthard et al., do not specifically disclose means for displaying a keystroke management window, wherein the keystroke management window is user modifiable to accept a local keystroke management setting, wherein if the local keystroke management setting is enabled, all hotkey keystrokes are processable at the remote computing device, wherein if the local keystroke management setting is disabled, all the hotkey keystrokes are processable at a first local computing device, and ***the remote computing device is configured to ignore*** the hotkey keystrokes, and that the user interface of the apparatus comprises a software feature configured to allow the user to select a connection icon from the desktop of the remote computing device; and wherein the remote computing device includes an operating system which does not allow the connection of the connection icon to be modified from the desktop by the user in absence of the software feature.

In the same field of endeavor, Henriquez discloses the claimed programmed computer apparatus, comprising, in the apparatus, a software feature configured to allow the user to select a connection icon from the desktop of the remote computing device, wherein the remote computing device includes an operating system which does

not allow the connection icon to be modified from the desktop by the user in absence of the software feature (Fig. 2 and paragraph 0036 that describes communication between a local system 210 and a remote system 220, wherein the local system can run an application residing on the remote system, and wherein the remote system sends icon information for the application to the local system, so that when a user selects an icon associated with the remote application on the local system, a remote desktop connection is established; further disclosing that at least a subset of the transmitted iconic connection data is persistent in the memory of the local system, and can be updated with later format versions, etc.; furthermore paragraphs 0006, 0030, and 0031 disclose that the remote computing device includes a Microsoft® Windows® operating system, e.g. Windows NT SERVER® operating system with Remote Desktop Protocol (RDP) feature); and

wherein selecting the connection icon by the user from the desktop allows a connection represented by the connection icon to become modifiable to alter the connection of the connection icon by the user (Figs. 3 and 6, paragraphs 0045 and 0050 which disclose the claimed apparatus, comprising icon data transmitted by a remote system being received by a local system, further disclosing that after decoding, the icon data is streamed into a file and the location of the icon file is noted in a regkey (registry key field), then disclosing that a shell (WINDOWS® operating system) is informed that an icon is ready to be displayed, when a specific file extension (e.g. .ico) is placed on the desktop, thereby disclosing placing the connection icon/link for the transmitted icon on the desktop, and

in Falcon et al. reference, Figs. 6-7 and column 7, lines 1-27 which show and disclose that when a user clicks on to select one of the connection icons (shown in Fig. 6 as "Office", "Work from Home", "MSN", and "New Connector"), a user interface with various tabs to modify the selected existing connection or to enter configuration data for a new connection is displayed, as shown in Fig. 7).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to provide, in the apparatus, a software feature configured to allow the user to select a connection icon from the desktop of the remote computing device, wherein selecting the connection icon by the user from the desktop allows a connection represented by the connection icon to become modifiable to alter the connection of the connection icon by the user, and wherein the remote computing device includes an operating system which does not allow the connection icon to be modified from the desktop by the user to alter the connection of the connection icon in absence of the software feature, as taught by Henriquez, in the apparatus of Falcon et al., as modified by Coulthard et al., because such an apparatus will provide a user-friendly interface for remote desktop applications, allowing users to easily modify the connection to applications executable on remote computers.

However, Falcon et al., as modified by Coulthard et al. and Henriquez, do not specifically disclose means for displaying a keystroke management window, wherein the keystroke management window is user modifiable to accept a local keystroke management setting, wherein if the local keystroke management setting is enabled, all hotkey keystrokes are processable at the remote computing device, wherein if the local

keystroke management setting is disabled, all the hotkey keystrokes are processable at a first local computing device, and ***the remote computing device is configured to ignore*** the hotkey keystrokes.

In the same field of endeavor, Perholtz et al. disclose the claimed apparatus, further comprising means for displaying a keystroke management window, wherein the keystroke management window is user modifiable to accept a local keystroke management setting, wherein if the local keystroke management setting is enabled, all hotkey keystrokes are processable at the remote computing device, wherein if the local keystroke management setting is disabled, all the hotkey keystrokes are processable at a first local computing device, and ***the remote computing device is configured to ignore*** the hotkey keystrokes (Flowchart of Fig. 7G, decision block 759 that checks for use of “Hot Keys”; paragraph 0288, lines 1-16 that disclose the use of “Hot Keys” for redirecting remote client’s input keystrokes/mouse data to the local host [i.e. by selecting, using hotkey keystrokes or mouse-clicks, from the Host menu, the option “Exit to Host mode”, that disables the local keystroke management setting, such that all the hotkey keystrokes are processed at the local computing device instead of at the remote computing device] and means to return back to the remote client’s normal mode of operation by tapping the left shift key three times within 2 seconds [a hotkey sequence that enables the local keystroke management setting, such that all the hotkey keystrokes are now processed back at the remote computing device]).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to provide code for displaying a keystroke

management window, wherein the keystroke management window is user modifiable to accept a local keystroke management setting, wherein if the local keystroke management setting is enabled, all hotkey keystrokes are processable at the remote computing device, wherein if the local keystroke management setting is disabled, all the hotkey keystrokes are processable at a first local computing device, and the remote computing device is configured to ignore the hotkey keystrokes, as taught by Perholtz et al., in the apparatus of Falcon et al., as modified by Coulthard et al. and Henriquez, so as to provide a user ability to use hot keys to execute applications at the local computing device as well as at the remote computing device, and be able to easily switch between them.

Consider **claim 33**, and **as it applies to claim 1 above**, Falcon et al., as modified by Coulthard et al., Henriquez and Perholtz et al., further show and disclose the claimed user interface, wherein when the remote computing device connects to the first local computing device, the desktop is operative to automatically display, directly on the desktop of the remote computing device, a plurality of applications stored and executable on the first local computing device (in Coulthard et al. reference, Fig. 11 that shows the Remote System Explorer desktop 1110 of the remote computing device (client) with a plurality of connection icons 1111-1113 corresponding to establishing connections with a plurality of remote systems 1120-1124, and a list of stored and executable applications 1130-1150 under each connection icon respectively; paragraph 0099 describes the same details).

Consider **claim 34**, and **as it applies to claim 7 above**, Falcon et al., as modified by Coulthard et al., Henriquez and Perholtz et al., further show and disclose the claimed method, further comprising the step of, when the remote computing device connects to the first local computing device, automatically displaying, directly on the desktop of the remote computing device, a plurality of applications stored and executable on the first local computing device (in Coulthard et al. reference, Fig. 11 that shows the Remote System Explorer desktop 1110 of the remote computing device (client) with a plurality of connection icons 1111-1113 corresponding to establishing connections with a plurality of remote systems 1120-1124, and a list of stored and executable applications 1130-1150 under each connection icon respectively; paragraph 0099 describes the same details).

Consider **claim 35**, and **as it applies to claim 17 above**, Falcon et al., as modified by Coulthard et al., Henriquez and Perholtz et al., further show and disclose the claimed ***non-transitory computer readable medium***, comprising: code for, when the remote computing device connects to the first local computing device, automatically displaying, directly on the desktop of the remote computing device, a plurality of applications stored and executable on the first local computing device (in Coulthard et al. reference, Fig. 11 that shows the Remote System Explorer desktop 1110 of the remote computing device (client) with a plurality of connection icons 1111-1113 corresponding to establishing connections with a plurality of remote systems 1120-

1124, and a list of stored and executable applications 1130-1150 under each connection icon respectively; paragraph 0099 describes the same details).

Consider **claim 36**, and **as it applies to claim 21 above**, Falcon et al., as modified by Coulthard et al., Henriquez and Perholtz et al., further show and disclose the claimed apparatus, comprising: means for, when the remote computing device connects to the first local computing device, automatically displaying, directly on the desktop of the remote computing device, a plurality of applications stored and executable on the first local computing device (in Coulthard et al. reference, Fig. 11 that shows the Remote System Explorer desktop 1110 of the remote computing device (client) with a plurality of connection icons 1111-1113 corresponding to establishing connections with a plurality of remote systems 1120-1124, and a list of stored and executable applications 1130-1150 under each connection icon respectively; paragraph 0099 describes the same details).

Consider **claim 37**, and **as it applies to claim 1 above**, Falcon et al., as modified by Coulthard et al., Henriquez and Perholtz et al., further show and disclose the claimed user interface, wherein selecting the connection icon by the user from the desktop allows a connection configuration of the connection icon to be displayed, allowing the user to edit or delete the connection of the connection icon (in Henriquez reference, Figs. 3 and 6, paragraphs 0045 and 0050 which disclose icon data transmitted by a remote system being received by a local system, further disclosing that

after decoding, the icon data is streamed into a file and the location of the icon file is noted in a regkey (registry key field), then disclosing that a shell (WINDOWS® operating system) is informed that an icon is ready to be displayed, when a specific file extension (e.g. .ico) is placed on the desktop, thereby disclosing placing the connection icon/link for the transmitted icon on the desktop, and

in Falcon et al. reference, Figs. 6-7 and column 7, lines 1-27 which show and disclose that when a user clicks on to select one of the connection icons (shown in Fig. 6 as "Office", "Work from Home", "MSN", and "New Connector"), a user interface with various tabs to modify the selected existing connection or to enter configuration data for a new connection is displayed, as shown in Fig. 7).

Consider **claim 38**, and **as it applies to claim 7 above**, Falcon et al., as modified by Coulthard et al., Henriquez and Perholtz et al., further show and disclose the claimed method, wherein selecting the connection icon by the user from the desktop allows a connection configuration of the connection icon to be displayed, allowing the user to edit or delete the connection of the connection icon (in Henriquez reference, Figs. 3 and 6, paragraphs 0045 and 0050 which disclose icon data transmitted by a remote system being received by a local system, further disclosing that after decoding, the icon data is streamed into a file and the location of the icon file is noted in a regkey (registry key field), then disclosing that a shell (WINDOWS® operating system) is informed that an icon is ready to be displayed, when a specific file extension (e.g. .ico)

is placed on the desktop, thereby disclosing placing the connection icon/link for the transmitted icon on the desktop, and

in Falcon et al. reference, Figs. 6-7 and column 7, lines 1-27 which show and disclose that when a user clicks on to select one of the connection icons (shown in Fig. 6 as "Office", "Work from Home", "MSN", and "New Connector"), a user interface with various tabs to modify the selected existing connection or to enter configuration data for a new connection is displayed, as shown in Fig. 7).

Consider **claim 39**, and **as it applies to claim 17 above**, Falcon et al., as modified by Coulthard et al., Henriquez and Perholtz et al., further show and disclose the claimed ***non-transitory computer readable medium***, wherein selecting the connection icon by the user from the desktop allows a connection configuration of the connection icon to be displayed, allowing the user to edit or delete the connection of the connection icon (in Henriquez reference, Figs. 3 and 6, paragraphs 0045 and 0050 which disclose icon data transmitted by a remote system being received by a local system, further disclosing that after decoding, the icon data is streamed into a file and the location of the icon file is noted in a regkey (registry key field), then disclosing that a shell (WINDOWS® operating system) is informed that an icon is ready to be displayed, when a specific file extension (e.g. .ico) is placed on the desktop, thereby disclosing placing the connection icon/link for the transmitted icon on the desktop, and

in Falcon et al. reference, Figs. 6-7 and column 7, lines 1-27 which show and disclose that when a user clicks on to select one of the connection icons (shown in Fig. 6 as

"Office", "Work from Home", "MSN", and "New Connector"), a user interface with various tabs to modify the selected existing connection or to enter configuration data for a new connection is displayed, as shown in Fig. 7).

Consider **claim 40**, and **as it applies to claim 21 above**, Falcon et al., as modified by Coulthard et al., Henriquez and Perholtz et al., further show and disclose the claimed apparatus, wherein selecting the connection icon by the user from the desktop allows a connection configuration of the connection icon to be displayed, allowing the user to edit or delete the connection of the connection icon (in Henriquez reference, Figs. 3 and 6, paragraphs 0045 and 0050 which disclose icon data transmitted by a remote system being received by a local system, further disclosing that after decoding, the icon data is streamed into a file and the location of the icon file is noted in a regkey (registry key field), then disclosing that a shell (WINDOWS® operating system) is informed that an icon is ready to be displayed, when a specific file extension (e.g. .ico) is placed on the desktop, thereby disclosing placing the connection icon/link for the transmitted icon on the desktop, and
in Falcon et al. reference, Figs. 6-7 and column 7, lines 1-27 which show and disclose that when a user clicks on to select one of the connection icons (shown in Fig. 6 as "Office", "Work from Home", "MSN", and "New Connector"), a user interface with various tabs to modify the selected existing connection or to enter configuration data for a new connection is displayed, as shown in Fig. 7).

Claims 4, 10, 25, 27 and 31 are rejected under 35 U.S.C. 103(a) as being unpatentable over **Falcon et al.** (**U.S. Patent Publication # 6,295,556 B1**) in view of **Coulthard et al. (U.S. Patent Application Publication # 2004/0003371 A1)** and further in view of **Henriquez (U.S. Patent Application Publication # 2004/0088377 A1)** and further in view of **Perholtz et al. (U.S. Patent Application Publication # 2002/0091850 A1)** and further in view of **Beadle et al. (U.S. Patent Publication # 7,039,709 B1).**

Consider **claim 4**, and **as it applies to claim 1 above**, Falcon et al., as modified by Coulthard et al., Henriquez and Perholtz et al., disclose the claimed user interface, except wherein the first connection icon and the second connection icon each includes a priority.

In the same field of endeavor, Beadle et al. disclose a user interface for managing a connection between a remote computing device and a local computing device, wherein the first connection icon and the second connection icon each includes a priority (Fig. 5A, “Select Default Server” block 507, “Override Defaults” block 511, and “Update Settings” button 515 that enable a user to set priorities in selecting different connections and other options; Fig. 6A that lists some of the options 601 that can be assigned priority values to arrive at the relative ratings 605; column 8, lines 28-33 that disclose the same details).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to provide a user interface, wherein the first connection

icon and the second connection icon each includes a priority, as taught by Beadle et al., in the user interface of Falcon et al., as modified by Coulthard et al., Henriquez and Perholtz et al., so as to allow users to assign different priorities to defined connections.

Consider **claim 10**, and **as it applies to claim 7 above**, Falcon et al., as modified by Coulthard et al., Henriquez and Perholtz et al., disclose the claimed method, except wherein the first connection icon and the second connection icon each includes a priority.

In the same field of endeavor, Beadle et al. disclose a method for managing a connection between a local computing device and a remote computing device, using a user interface, wherein the first connection icon and the second connection icon each includes a priority (Fig. 5A, “Select Default Server” block 507, “Override Defaults” block 511, and “Update Settings” button 515 that enable a user to set priorities in selecting different connections and other options; Fig. 6A that lists some of the options 601 that can be assigned priority values to arrive at the relative ratings 605; column 8, lines 28-33 that disclose the same details).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to provide a method for a user interface, wherein the first connection icon and the second connection icon each includes a priority, as taught by Beadle et al., in the method of Falcon et al., as modified by Coulthard et al., Henriquez and Perholtz et al., so as to allow users to assign different priorities to defined connections.

Consider **claim 25**, and **as it applies to claim 1 above**, Falcon et al., as modified by Coulthard et al., Henriquez and Perholtz et al., further disclose the claimed user interface for managing a connection between a remote computing device and a local computing device, wherein selecting the first connection icon allows the user to edit or delete the first connection (in Falcon et al. reference, Fig. 7 which shows a second interface to manage connections by setting different configuration options for a connection; column 2, lines 7-9 disclose the same details).

However, Falcon et al., as modified by Coulthard et al., Henriquez and Perholtz et al., do not specifically disclose the user interface wherein the remote computing device is a thin client, and wherein the user interface is to be displayed at the thin client.

In the same field of endeavor, Beadle et al. disclose the claimed user interface for managing a connection between a remote computing device and a local computing device, wherein the remote computing device is a thin client (column 1, lines 32-34 which disclose that clients can be “dumber” systems (thin clients) adapted for limited use with a network); and

wherein the user interface is to be displayed at the thin client (column 2, lines 54-57 that disclose a graphical user interface for receiving user selection at the remote thin client, and a connection utility for connecting the client with a selected local server).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to provide a user interface, wherein the remote computing device is a thin client, and wherein the user interface is to be displayed at the

thin client, as taught by Beadle et al., in the user interface of Falcon et al., as modified by Coulthard et al., Henriquez and Perholtz et al., so as to provide support for connection management to clients with thin remote devices.

Consider **claim 27**, and **as it applies to claim 17 above**, Falcon et al., as modified by Coulthard et al., Henriquez and Perholtz et al., disclose the claimed ***non-transitory computer readable medium storing*** computer-executable program code, wherein selecting the first connection icon allows the user to edit or delete the first connection (in Falcon et al. reference, Fig. 7 which shows a second interface to manage connections by setting different configuration options for a connection; column 2, lines 7-9 disclose the same details); and

wherein the first application is different from the second application (in Coulthard et al. reference, Fig. 11, that shows three different connections 1111-1113 between a Developer 1 and three Remote Systems 1120, 1122 and 1124, wherein connection 1111 provides access to Tool A (application 1130) and connection 1112 provides access to Tool C (application 1150); paragraph 0099 describes the same details).

However, Falcon et al., as modified by Coulthard et al., Henriquez and Perholtz et al., do not specifically disclose that the remote computing device is a thin client, wherein the user interface is to be displayed at the thin client, and wherein the second connection is between the thin client and a second local computing device.

In the same field of endeavor, Beadle et al. disclose the claimed computer-executable program code for managing a connection between a remote computing

device and a local computing device, wherein the remote computing device is a thin client (column 1, lines 32-34 which disclose that clients can be “dumber” systems (thin clients) adapted for limited use with a network);
wherein the user interface is to be displayed at the thin client (column 2, lines 54-57 that disclose a graphical user interface for receiving user selection at the remote thin client, and a connection utility for connecting the client with a selected local server); and
wherein the second connection is between the thin client and a second local computing device (Fig. 10 that shows a second connection using modem transmission; column 10, lines 6-24 which disclose a first connection via satellite to a DirectPC application and a second modem connection to a server for a financial application).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to provide computer-executable program code, wherein the remote computing device is a thin client, wherein the user interface is to be displayed at the thin client, and wherein the second connection is between the thin client and a second local computing device, as taught by Beadle et al., in the computer-executable program code of Falcon et al., as modified by Coulthard et al., Henriquez and Perholtz et al., so as to provide support for connection management to clients with thin remote devices.

Consider **claim 31**, and **as it applies to claim 17 above**, Falcon et al., as modified by Coulthard et al., Henriquez and Perholtz et al., disclose the claimed ***non-transitory computer readable medium storing*** computer-executable program code,

except wherein the first connection icon and the second connection icon each includes a priority.

In the same field of endeavor, Beadle et al. disclose the ***non-transitory computer readable medium storing*** computer-executable program code, wherein the first connection icon and the second connection icon each includes a priority (claims 10-12; Fig. 5A, "Select Default Server" block 507, "Override Defaults" block 511, and "Update Settings" button 515 that enable a user to set priorities in selecting different connections and other options; Fig. 6A that lists some of the options 601 that can be assigned priority values to arrive at the relative ratings 605; column 8, lines 28-33 that disclose the same details).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to provide computer-executable program code, wherein the first connection icon and the second connection icon each includes a priority, as taught by Beadle et al., in the computer-executable program code of Falcon et al., as modified by Coulthard et al., Henriquez and Perholtz et al., so as to allow users to assign different priorities to defined connections.

Claim 5 is rejected under 35 U.S.C. 103(a) as being unpatentable over **Falcon et al.** (**U.S. Patent Publication # 6,295,556 B1**) in view of **Coulthard et al. (U.S. Patent Application Publication # 2004/0003371 A1)** and further in view of **Henriquez (U.S. Patent Application Publication # 2004/0088377 A1)** and further in view of

Perholtz et al. (U.S. Patent Application Publication # 2002/0091850 A1) and further in view of **Lele (U.S. Patent Publication # 7,181,524 B1).**

Consider **claim 5**, and **as it applies to claim 1 above**, Falcon et al., as modified by Coulthard et al., Henriquez and Perholtz et al., disclose the claimed user interface, except wherein the priority is a failover order.

In the same field of endeavor, Lele discloses a user interface, wherein the priority is a failover order (column 1, lines 21-27 that disclose a plurality of servers connected in a server cluster to provide failover redundancy; Fig. 1, Rules block 154 and Selection Algorithm block 155 that specify server selection criteria; thereby disclosing server failover order that a user may specify as a priority option in the connection management).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to provide a user interface for managing a connection between a remote computing device and a local computing device, wherein the priority is a failover order, as taught by Lele, in the user interface of Falcon et al., as modified by Coulthard et al., Henriquez and Perholtz et al., so as to provide an alternate connection path to a server, in case the selected server fails.

Claims 6, 12, 20 and 24 are rejected under 35 U.S.C. 103(a) as being unpatentable over **Falcon et al.** (**U.S. Patent Publication # 6,295,556 B1**) in view of **Coulthard et al. (U.S. Patent Application Publication # 2004/0003371 A1)** and

further in view of **Henriquez (U.S. Patent Application Publication # 2004/0088377 A1)** and further in view of **Perholtz et al. (U.S. Patent Application Publication # 2002/0091850 A1)** and further in view of **Ritchy et al. (U.S. Patent Application Publication # 2004/0183831 A1).**

Consider **claim 6**, and **as it applies to claim 1 above**, Falcon et al., as modified by Coulthard et al., Henriquez and Perholtz et al., disclose the claimed user interface, except further comprising a desktop shell window, wherein the desktop shell window is modifiable at run-time by the user at the remote computing device to accept a desktop shell setting, wherein if the desktop shell setting is disabled, an alternate user interface is selected and the user interface is disabled.

In the same field of endeavor, Ritchy et al. disclose a desktop window, wherein the desktop shell window is modifiable at run-time by the user at the remote computing device to accept a desktop shell setting, wherein if the desktop shell setting is disabled, an alternate user interface is selected and the user interface is disabled (Fig. 9 that shows a default desktop window and a pull-down to select alternate desktop shell if the user so desires; paragraph 0049, lines 9-11 which disclose that different shells for the desktop are selectable in the Property Editor window, and portal administrators and end users can also change a desktop's shell, thereby disclosing that the desktop shell window is modifiable at run-time by the user at the remote computing device to accept a desktop shell setting; wherein if the desktop shell setting is disabled, an alternate user interface is selected and the user interface is disabled).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to provide a desktop shell window, wherein the desktop shell window is modifiable at run-time by the user at the remote computing device to accept a desktop shell setting, wherein if the desktop shell setting is disabled, an alternate user interface is selected and the user interface is disabled, as taught by Ritchy et al., in the user interface of Falcon et al., as modified by Coulthard et al., Henriquez and Perholtz et al., so as to provide multiple operating systems environments for the user to choose from, based on user's preferences, on the same desktop.

Consider **claim 12**, and **as it applies to claim 7 above**, Falcon et al., as modified by Coulthard et al., Henriquez and Perholtz et al., disclose the claimed method, except further comprising the steps of displaying a desktop shell window, wherein the desktop shell window is modifiable at run-time by a user at the remote computing device to accept a desktop shell setting; selecting an alternate user interface, if the desktop shell setting is disabled; disabling the user interface, if the desktop shell setting is disabled.

In the same field of endeavor, Ritchy et al. disclose a desktop window, wherein the desktop shell window is modifiable at run-time by a user at the remote computing device to accept a desktop shell setting; selecting an alternate user interface, if the desktop shell setting is disabled; disabling the user interface, if the desktop shell setting is disabled (Fig. 9 that shows a default desktop window and a pull-down to select alternate desktop shell if the user so desires; paragraph 0049, lines 9-11 which disclose

that different shells for the desktop are selectable in the Property Editor window, and portal administrators and end users can also change a desktop's shell, thereby disclosing that the desktop shell window is modifiable at run time by the user at the remote computing device to accept a desktop shell setting, selecting an alternate user interface, if the desktop shell setting is disabled, and disabling the user interface, if the desktop shell setting is disabled).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to provide a desktop shell window, wherein the desktop shell window is modifiable at run-time by a user at the remote computing device to accept a desktop shell setting; selecting an alternate user interface, if the desktop shell setting is disabled; disabling the user interface, if the desktop shell setting is disabled, as taught by Ritchy et al., in the method of Falcon et al., as modified by Coulthard et al., Henriquez and Perholtz et al., so as to provide multiple operating systems environments for the user to choose from, based on user's preferences, on the same desktop.

Consider **claim 20**, and **as it applies to claim 17 above**, Falcon et al., as modified by Coulthard et al., Henriquez and Perholtz et al., disclose the claimed ***non-transitory computer readable medium storing*** computer-executable program code, except said program code comprising code for displaying a desktop shell window, wherein the desktop shell window is modifiable at run-time by a user at the remote computing device to accept a desktop shell setting; code for selecting an alternate user

interface, if the desktop shell setting is disabled; and code for disabling the user interface, if the desktop shell setting is disabled.

In the same field of endeavor, Ritchy et al. disclose a ***non-transitory computer readable medium storing*** computer-executable program code, said program comprising code for permitting the computer to perform a step for displaying a desktop shell window, wherein the desktop shell window is modifiable at run-time by a user at the remote computing device to accept a desktop shell setting; a selecting step for selecting an alternate user interface, if the desktop shell setting is disabled; a disabling step for disabling the user interface, if the desktop shell setting is disabled (Claims 20-38, 60-80, and 101-120; that shows a default desktop window with a user interface (pull-down) to select an alternate desktop shell if the user so desires; paragraph 0049, lines 9-11 which disclose that different shells for the desktop are selectable in the Property Editor window, and portal administrators and end users can also change a desktop's shell, thereby disclosing that the desktop shell window is modifiable at run-time by a user at the remote computing device to accept a desktop shell setting, selecting an alternate user interface, if the desktop shell setting is disabled, and disabling the user interface, if the desktop shell setting is disabled).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to provide a computer-readable storage medium with stored program code for managing a connection between a local computing device and a remote computing device, said program comprising code for permitting the computer to perform a step for displaying a desktop shell window, wherein the desktop shell

window is modifiable at run-time by a user at the remote computing device to accept a desktop shell setting; a selecting step for selecting an alternate user interface, if the desktop shell setting is disabled; a disabling step for disabling the user interface, if the desktop shell setting is disabled, as taught by Ritchy et al., in the computer-executable program code of Falcon et al., as modified by Coulthard et al., Henriquez and Perholtz et al., so as to provide a user ability to select any one of the many available desktop shells that is most suited to the user.

Consider **claim 24**, and **as it applies to claim 21 above**, Falcon et al., as modified by Coulthard et al., Henriquez and Perholtz et al., show and disclose the claimed programmed computer apparatus, except further comprising means for displaying a desktop shell window, wherein the desktop shell window is modifiable at run-time by a user at the remote computing device; means for selecting an alternate user interface, if the desktop shell setting is disabled; and means for disabling the user interface, if the desktop shell setting is disabled.

In the same field of endeavor, Ritchy et al. show and disclose the claimed programmed computer apparatus, further comprising means for displaying a desktop shell window, wherein the desktop shell window is modifiable at run time by a user at the remote computing device; means for selecting an alternate user interface, if the desktop shell setting is disabled; and means for disabling the user interface, if the desktop shell setting is disabled (Fig. 9 that shows a default desktop window and a pull-down to select alternate desktop shell if the user so desires; paragraph 0049, lines 9-11

which disclose that different shells for the desktop are selectable in the Property Editor window, and portal administrators and end users can also change a desktop's shell, thereby disclosing that the desktop shell window is modifiable at run-time by the user at the remote computing device to accept a desktop shell setting, selecting an alternate user interface, if the desktop shell setting is disabled, and disabling the improved user interface, if the desktop shell setting is disabled).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to provide means for displaying a desktop shell window, wherein the desktop shell window is modifiable at run time by a user at the remote computing device; means for selecting an alternate user interface, if the desktop shell setting is disabled; and means for disabling the user interface, if the desktop shell setting is disabled, as taught by Ritchy et al., in the programmed computer apparatus of Falcon et al., as modified by Coulthard et al., Henriquez and Perholtz et al., so as to provide a user ability to select any one of the many available desktop shells that is most suited to the user.

Claims 11 and 32 are rejected under 35 U.S.C. 103(a) as being unpatentable over **Falcon et al.** (**U.S. Patent Publication # 6,295,556 B1**) in view of **Coulthard et al. (U.S. Patent Application Publication # 2004/0003371 A1)** and further in view of **Henriquez (U.S. Patent Application Publication # 2004/0088377 A1)** and further in view of **Perholtz et al. (U.S. Patent Application Publication # 2002/0091850 A1)** and

further in view of **Beadle et al. (U.S. Patent Publication # 7,039,709 B1)** and further in view of **Lele (U.S. Patent Publication # 7,181,524 B1).**

Consider **claim 11**, and **as it applies to claim 10 above**, Falcon et al., as modified by Coulthard et al., Henriquez, Perholtz et al. and Beadle et al., disclose the claimed method, except wherein the priority is a failover order.

In the same field of endeavor, Lele discloses the claimed method, wherein the priority is a failover order (column 1, lines 21-27 that disclose a plurality of servers connected in a server cluster to provide failover redundancy; Fig. 1, Rules block 154 and Selection Algorithm block 155 that specify server selection criteria; thereby disclosing server failover order that a user may specify as a priority option in the connection management).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to provide a method for managing a connection between a remote computing device and a local computing device using a user interface, wherein the priority is a failover order, as taught by Lele, in the method of Falcon et al., as modified by Coulthard et al., Henriquez, Perholtz et al., and Beadle et al., so as to provide an alternate connection path to a server, in case the selected server fails.

Consider **claim 32**, and **as it applies to claim 31 above**, Falcon et al., as modified by Coulthard et al., Henriquez, Perholtz et al. and Beadle et al., disclose the

claimed ***non-transitory computer readable medium storing*** computer-executable program code, except wherein the priority is a failover order.

In the same field of endeavor, Lele discloses the claimed ***non-transitory computer readable medium storing*** computer-executable program code, wherein the priority is a failover order (column 1, lines 21-27 that disclose a plurality of servers connected in a server cluster to provide failover redundancy; Fig. 1, Rules block 154 and Selection Algorithm block 155 that specify server selection criteria; thereby disclosing server failover order that a user may specify as a priority option in the connection management).

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to provide computer-executable program code for managing a connection between a remote computing device and a local computing device, wherein the priority is a failover order, as taught by Lele, in the computer-executable program code of Falcon et al., as modified by Coulthard et al., Henriquez, Perholtz et al and Beadle et al., so as to provide an alternate connection path to a server, in case the selected server fails.

Response to Arguments

Applicants' arguments with respect to claims 1, 4-7, 10-12, 17, 20-21 and 24-25, 27, and 31-40, filed 06/28/2011 have been fully considered but they are not persuasive. After carefully reviewing the arguments and the prior art used to reject the claims, the examiner has concluded that the applied prior art teach/suggest each and every

element of the presented claims. The claims are, therefore, obvious over the cited prior art, non-novel, and not allowable in their present form. The examiner's response to presented arguments is shown below:

The examiner has carefully reviewed the **independent claims 1, 7, 17 and 21,** and based on the amendments, withdrawn the 35 USC 112, first paragraph as well as 35 USC 112, second paragraph rejections for these claims and their respective **dependent claims 4-6, 10-12, 20, 24-25, 27 and 31-40**. However, the 35 USC 112, second paragraph rejections for **claims 21, 24, 36 and 40** that include "means for" phrases, are maintained, because the paragraphs 0044-0046 cited by the applicants in their arguments describe the hardware and software components of any general purpose computer, and not the specific structures, components and algorithms specifically applicable to the applicants invention.

Next, consider the 35 USC 103(a) rejections for the amended **independent claims 1, 7, 17, and 21,** of which claim 1 is a representative claim. On page 19 of the "Remarks" section, the applicants argue that in the cited Coulthard reference, the tools mentioned by Coulthard cannot be equated with "an application available for execution on the first local computing device", as specified in claim 1, because, in Coulthard, the tools reside and execute on a remote computing device, not on a local computing device as claimed.

The examiner respectfully disagrees with this argument. The reason for the applicants' misinterpretation of Coulthard's teachings is due to the difference in nomenclature for labeling local and remote computing devices. Whereas the applicants

name the server 120 of Fig. 1 [where the applications execute] as a local computing device, Coulthard names such servers 1120-1124 in Fig. 11, as remote computing systems. Therefore, the examiner's mapping of the computing devices in the previous office action is still valid and the cited Coulthard reference still reads on the recited text of claim 1.

On page 20 of the "Remarks" section, the applicants further argue that the cited Perholtz reference does not disclose the claimed feature "if the local keystroke management setting is disabled, ***all the hotkey keystrokes are processed at a first local computing device and the remote computing device is configured to ignore the hotkey keystrokes at the remote computing device",*** of claim 1. The examiner begs to differ with this argument. The only one specific example cited for this argument is the keystroke combination that switches the computing device between a local computing mode or a remote computing mode, which may not be considered as a hotkey keystroke function, but a mode switch function, that will be needed in the applicants' invention as well.

In conclusion, the prior art selected by the examiner does adequately teach each and every claim element of claims 1, 4-7, 10-12, 17, 20-21, 24-25, 27 and 31-40, which are, therefore, obvious over the cited prior art, non-novel, and not in the condition for allowance in their present form. **Claims 1, 4-7, 10-12, 17, 20-21, 24-25, 27 and 31-40 remain rejected.**

Conclusion

THIS ACTION IS MADE FINAL. Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

Any response to this Office Action should be **faxed to** (571) 273-8300 **or mailed to**:

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Art Unit: 2443

Hand-delivered responses should be brought to

Customer Service Window
Randolph Building
401 Dulany Street
Alexandria, VA 22314

Any inquiry concerning this communication or earlier communications from the

Examiner should be directed to Kishin G. Belani whose telephone number is (571) 270-1768. The Examiner can normally be reached on Monday-Friday from 6:00 am to 5:00 pm.

If attempts to reach the Examiner by telephone are unsuccessful, the Examiner's supervisor, Tonia Dollinger can be reached on (571) 272-4170. The fax phone number for the organization where this application or proceeding is assigned is (571) 273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free) or 703-305-3028.

Any inquiry of a general nature or relating to the status of this application or proceeding should be directed to the receptionist/customer service whose telephone number is (571) 272-0800.

/K. G. B./
Examiner, Art Unit 2443

September 9, 2011

/PHUOC NGUYEN/
Primary Examiner, Art Unit 2443